Introduction

DataFrame veya Series'teki verileri, bir sütunun tamamı için her seferinde bir değer olmak üzere, **map** fonksiyonuyla dönüştürebiliriz. Ancak, çoğu zaman verilerimizi gruplamak ve ardından verinin içinde bulunduğu gruba özel bir işlem yapmak isteriz.

### groupby() ile Veri Gruplama

Öğreneceğiniz üzere, bunu **groupby()** operasyonuyla yaparız. Ayrıca, DataFrame'lerinizi indekslemenin daha karmaşık yolları ve verilerinizi nasıl sıralayacağınız gibi ek konuları da ele alacağız.

# Groupwise analysis[¶](https://www.kaggle.com/code/residentmario/grouping-and-sorting" \l "Groupwise-analysis" \t "_self)

Şimdiye kadar sıklıkla kullandığımız fonksiyonlardan biri **value\_counts()** fonksiyonudur. **value\_counts()**'ın yaptığı işi, aşağıdakileri yaparak tekrarlayabiliriz:

import pandas as pd

reviews = pd.read\_csv("../input/wine-reviews/winemag-data-130k-v2.csv", index\_col=0)

pd.set\_option("display.max\_rows", 5)

reviews.groupby('points').points.count()

points

80 397

81 692

...

99 33

100 19

Name: points, Length: 21, dtype: int64

**groupby()**, aynı puan değerini alan şarapların bir grubunu oluşturdu. Ardından, bu grupların her biri için **points()** sütununu alıp kaç kez göründüğünü saydık. **value\_counts()** aslında bu **groupby()** operasyonunun bir kısaltmasıdır.

### Gruplama ile Özetleme

Daha önce kullandığımız özetleme fonksiyonlarının herhangi birini bu verilerle birlikte kullanabiliriz. Örneğin, her puan kategorisindeki en ucuz şarabı bulmak için aşağıdakileri yapabiliriz:

reviews.groupby('points').price.min()

,

points

80 5.0

81 5.0

...

99 44.0

100 80.0

Name: price, Length: 21, dtype: float64

Oluşturduğumuz her grubu, sadece eşleşen değerlere sahip verileri içeren DataFrame'imizin bir kesiti olarak düşünebilirsiniz. Bu DataFrame'e doğrudan **apply()** metodu ile erişebilir ve veriyi istediğimiz şekilde işleyebiliriz.

### apply() Metodu ile Veri Manipülasyonu

Örneğin, veri setindeki her şaraphaneden incelenen ilk şarabın adını seçmenin bir yolu aşağıdadır:

reviews.groupby('winery').apply(lambda df: df.title.iloc[0])

winery

1+1=3 1+1=3 NV Rosé Sparkling (Cava)

10 Knots 10 Knots 2010 Viognier (Paso Robles)

...

àMaurice àMaurice 2013 Fred Estate Syrah (Walla Walla V...

Štoka Štoka 2009 Izbrani Teran (Kras)

Length: 16757, dtype: object

Daha ayrıntılı kontrol için, birden fazla sütuna göre de gruplama yapabilirsiniz. Örneğin, ülkeye **ve** eyalete göre en iyi şarabı nasıl seçeceğimiz aşağıda gösterilmiştir:

reviews.groupby(['country', 'province']).apply(lambda df: df.loc[df.points.idxmax()])

| country | description | designation | points | price | province | region\_1 | region\_2 | taster\_name | taster\_twitter\_handle | title | variety | winery |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| country | province |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Argentina | Mendoza Province | Argentina | If the color doesn't tell the full story, the ... | Nicasia Vineyard | 97 | 120.0 | Mendoza Province | Mendoza | NaN | Michael Schachner | @wineschach | Bodega Catena Zapata 2006 Nicasia Vineyard Mal... | Malbec | Bodega Catena Zapata |
| Other | Argentina | Take note, this could be the best wine Colomé ... | Reserva | 95 | 90.0 | Other | Salta | NaN | Michael Schachner | @wineschach | Colomé 2010 Reserva Malbec (Salta) | Malbec | Colomé |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| Uruguay | San Jose | Uruguay | Baked, sweet, heavy aromas turn earthy with ti... | El Preciado Gran Reserva | 87 | 50.0 | San Jose | NaN | NaN | Michael Schachner | @wineschach | Castillo Viejo 2005 El Preciado Gran Reserva R... | Red Blend | Castillo Viejo |
| Uruguay | Uruguay | Cherry and berry aromas are ripe, healthy and ... | Blend 002 Limited Edition | 91 | 22.0 | Uruguay | NaN | NaN | Michael Schachner | @wineschach | Narbona NV Blend 002 Limited Edition Tannat-Ca... | Tannat-Cabernet Franc | Narbon |

**groupby()**'a değinmeye değer bir diğer metot ise **agg()**'dir. Bu metot, DataFrame'iniz üzerinde aynı anda birden fazla farklı fonksiyon çalıştırmanıza olanak tanır. Örneğin, veri setinin basit bir istatistiksel özetini aşağıdaki gibi oluşturabiliriz:

reviews.groupby(['country']).price.agg([len, min, max])

| len | min | max |
| --- | --- | --- |
| country |  |  |  |
| Argentina | 3800 | 4.0 | 230.0 |
| Armenia | 2 | 14.0 | 15.0 |
| ... | ... | ... | ... |
| Ukraine | 14 | 6.0 | 13.0 |
| Uruguay | 109 | 10.0 | 130.0 |

groupby()'nin etkili kullanımı, veri kümenizle birçok güçlü şey yapmanıza olanak tanır.

# Multi-indexes[¶](https://www.kaggle.com/code/residentmario/grouping-and-sorting" \l "Multi-indexes" \t "_self)

Şimdiye kadar gördüğümüz tüm örneklerde, tek etiketli bir indekse sahip **DataFrame** veya **Series** nesneleriyle çalıştık. **groupby()** ise biraz farklıdır çünkü yaptığımız işleme bağlı olarak bazen **"multi-index"** adı verilen bir sonuç oluşturur.

### Multi-Index Nedir?

**Multi-index**, birden fazla seviyeye sahip olmasıyla normal bir indeksten ayrılır. Örneğin:

countries\_reviewed = reviews.groupby(['country', 'province']).description.agg([len])

countries\_reviewed

| len |
| --- |
| country | province |  |
| Argentina | Mendoza Province | 3264 |
| Other | 536 |
| ... | ... | ... |
| Uruguay | San Jose | 3 |
| Uruguay | 24 |

mi = countries\_reviewed.index

type(mi)

pandas.core.indexes.multi.MultiIndex

Multi-index yapılarının, tek seviyeli indekslerde bulunmayan, kendi katmanlı yapılarıyla başa çıkmak için çeşitli metotları vardır. Ayrıca, bir değere erişmek için iki seviyeli etikete ihtiyaç duyarlar. Multi-index çıktısıyla uğraşmak, pandas'a yeni başlayan kullanıcılar için yaygın bir "küçük ama önemli bir hata" kaynağıdır.

### MultiIndex Kullanımı

Multi-index'lerin kullanım durumları ve onlarla nasıl çalışılacağına dair talimatlar, **pandas** dokümantasyonunun **MultiIndex / Advanced Selection** bölümünde ayrıntılı olarak yer almaktadır.

Ancak, genellikle en sık kullanacağınız multi-index metodu, onu normal bir indekse geri dönüştürmek için kullanılan **reset\_index()** metodudur:

countries\_reviewed.reset\_index()

| country | province | len |
| --- | --- | --- |
| 0 | Argentina | Mendoza Province | 3264 |
| 1 | Argentina | Other | 536 |
| ... | ... | ... | ... |
| 423 | Uruguay | San Jose | 3 |
| 424 | Uruguay | Uruguay | 24 |

# Sorting[¶](https://www.kaggle.com/code/residentmario/grouping-and-sorting" \l "Sorting" \t "_self)

**countries\_reviewed**'a tekrar baktığımızda, gruplamanın verileri değer sırasına göre değil, indeks sırasına göre döndürdüğünü görebiliriz. Yani, bir **groupby** işleminin sonucunu çıktı olarak alırken, satırların sırası verideki değerlere değil, indeksteki değerlere bağlıdır.

İstediğimiz sırada verileri elde etmek için, bunları kendimiz sıralayabiliriz. **sort\_values()** metodu bu iş için oldukça kullanışlıdır.

countries\_reviewed = countries\_reviewed.reset\_index()

countries\_reviewed.sort\_values(by='len')

| country | province | len |
| --- | --- | --- |
| 179 | Greece | Muscat of Kefallonian | 1 |
| 192 | Greece | Sterea Ellada | 1 |
| ... | ... | ... | ... |
| 415 | US | Washington | 8639 |
| 392 | US | California | 36247 |

**sort\_values()** metodu, varsayılan olarak artan (ascending) bir sıralama yapar; yani en düşük değerler önce gelir. Ancak çoğu zaman, yüksek sayıların önce geldiği azalan (descending) bir sıralama isteriz. Bu da şu şekilde yapılır:

countries\_reviewed.sort\_values(by='len', ascending=False)

| country | province | len |
| --- | --- | --- |
| 392 | US | California | 36247 |
| 415 | US | Washington | 8639 |
| ... | ... | ... | ... |
| 63 | Chile | Coelemu | 1 |
| 149 | Greece | Beotia | 1 |

İndeks değerlerine göre sıralama yapmak için, **sort\_index()** adında benzer bir metot kullanılır. Bu metodun argümanları ve varsayılan sıralama düzeni **sort\_values()** ile aynıdır:

countries\_reviewed.sort\_index()

| country | province | len |
| --- | --- | --- |
| 0 | Argentina | Mendoza Province | 3264 |
| 1 | Argentina | Other | 536 |
| ... | ... | ... | ... |
| 423 | Uruguay | San Jose | 3 |
| 424 | Uruguay | Uruguay | 24 |

Son olarak, aynı anda birden fazla sütuna göre sıralama yapabileceğinizi bilin:

countries\_reviewed.sort\_values(by=['country', 'len'])

| country | province | len |
| --- | --- | --- |
| 1 | Argentina | Other | 536 |
| 0 | Argentina | Mendoza Province | 3264 |
| ... | ... | ... | ... |
| 424 | Uruguay | Uruguay | 24 |
| 419 | Uruguay | Canelones | 43 |